Reguläre Ausdrücke / Regular Expressions („RegEx“)

Reguläre Ausdrücke (*engl.:* Regular Expressions, *Abk.:* RegEx) sind ein mächtiges Werkzeug zur Verarbeitung von Zeichenketten. Sie kommen in vielen Programmiersprachen und Filterprogrammen zum Einsatz.

Zeichen

x	das Zeichen „x“
\t	Horizontal-Tabulator (Tab)
\n	Zeilenumbruch, neue Zeile (New Line, Line Feed, LF)
\r	Schlittenrücklauf (Carriage Return, CR)
\x*hh*	Zeichen mit Hexadezimalwert *hh*
\u*hhhh*	Unicodezeichen mit Hexadezimalwert *hhhh*
****	hebt ggf. die Metabedeutung des nächsten Zeichens auf
****	Rückwärtsschrägstrich (Blackslash) „\“
\(	das Zeichen „(“
\[	das Zeichen „[“
\{	das Zeichen „{“
\.	das Punkt-Zeichen „.“
\^	das Zirkumflex-Zeichen „^“
\$	das Dollar-Zeichen „$“

Zeichenklassen

[abc]	eines der Zeichen „a“, „b“ und „c“
[^abc]	alles außer eines der Zeichen „a“, „b“ und „c“
[A-Z]	eines der Zeichen: „A“ bis „Z“
[A-Za-z]	eines die Zeichen „A“ bis „Z“ und „a“ bis „z“
[a-z&&[^st]]	eines der Zeichen „a“ bis „z“ außer die Zeichen „s“ und „t“

Vordefinierte Zeichenklassen

.	ein beliebiges Zeichen
\d	eine Dezimalziffer `[0-9]`
\D	ein Zeichen, das keine Dezimalziffer ist
\s	ein Leerraumzeichen (Whitespace) `[ \t\n\x08\x0c\r]`
\S	ein Zeichen, das kein Leerraumzeichen ist
\w	ein Wortzeichen `[_A-Za-z0-9]`
\W	ein Zeichen, das kein Wortzeichen ist

POSIX-Zeichenklassen

\p{Upper}	ein ASCII-Großbuchstabe `[A-Z]`
\p{Lower}	ein ASCII-Kleinbuchstabe `[a-z]`
\p{ASCII}	ein ASCII-Zeichen `[\x00-\x7f]`
\p{Alpha}	ein ASCII-Buchstabe `[A-Za-z]`
\p{Digit}	eine ASCII-Dezimalziffer `[0-9]`
\p{Alnum}	ein ASCII Buchstabe oder eine ASCII-Dezimalziffer `[A-Za-z0-9]`

Unicode-Zeichenklassen

\p{Lu}	ein Unicode-Großbuchstabe
\p{Ll}	ein Unicode-Kleinbuchstabe
\p{Sc}	ein Unicode-Währungssymbol
\p{Nl}	ein Unicode-Buchstabe oder eine Unicode-Dezimalziffer

Grenzen

^	sucht vom Zeilenanfang an
$	sucht bis Zeilenende
\<	leere Zeichenkette am Wortanfang
\>	leere Zeichenkette am Wortende
\b	leere Zeichenkette am Wortanfang oder am Wortende
\B	leere Zeichenkette, die nicht den Anfang oder das Ende eines Wortes bildet
\A	Anfang der Eingabe
\z	Ende der Eingabe

Wiederholungen

?	vorheriges Zeichen darf kein mal oder einmal wiederholt sein (einmal oder gar nicht) z.B.: `a?`
*****	vorheriges Zeichen darf beliebig oft wiederholt sein, oder gar nicht vorkommen (mehrmals oder gar nicht) z.B.: `a*`
+	vorheriges Zeichen darf beliebig oft wiederholt sein, soll aber mindestens einmal vorkommen (mindestens einmal) z.B.: `a+`
{n}	exakt n-mal z.B.: `a{3}`
{n,}	mindestens n-mal z.B.: `a{3,}`
{n,m}	n- bis m-mal z.B.: `a{3,7}`
{0,n}	maximal n-mal z.B.: `a{0,7}`

Logische Operatoren

XY	X gefolgt von Y
X|Y	X oder Y
(X)	X als Gruppe
(?:X)	X als Gruppe ohne spätere Bezugsmöglichkeit
X(?=Y)	auf X muß Y folgen
X(?!Y)	auf X darf kein Y folgen
(?<=X)Y	X muß vor Y stehen
(?<!X)Y	X darf nicht vor Y stehen
(?(B)W|F)	Wenn die gegebene „Bedingung“ B gefunden wird, kommt der „wahr-Ausdruck“ W zur Anwendung, andernfalls kommt der „falsch-Ausdruck“ F zur Anwendung.

Signale (Flags)

(?i)	Groß-/Kleinschreibung unabhängig (ASCII)
(?iu)	Groß-/Kleinschreibung unabhängig (Unicode)
(?m)	mehrzeiliger Modus
(?s)	Einzelzeilen- / Punktall-Modus

Hinweise (für den mehrzeiligen Modus)

\n	ein Zeilenumbruch im Unix/Linux-Format
\r	ein Zeilenumbruch im alten MacOS-Format (bis1996)
\n\r	ein Zeilenumbruch im MacOS-Format
\r\n	ein Zeilenumbruch im DOS- und Windows-Format

Links:

`jex.im` `regviz.org`
`debuggex.com` `txt2re.com`
`myregextester.com`
`youtu.be/28o8MhvCFGI`